UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/509,030	07/18/2005	Zhaochang Xu	4533-0112PUS1	1536

2292 7590 03/09/2010

BIRCH STEWART KOLASCH & BIRCH
PO BOX 747
FALLS CHURCH, VA 22040-0747

EXAMINER	
PETRANEK, JACOB ANDREW	
ART UNIT	PAPER NUMBER
2183	

NOTIFICATION DATE	DELIVERY MODE
03/09/2010	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

mailroom@bskb.com

Office Action Summary	Application No.	Applicant(s)	
	10/509,030	XU ET AL.	
	Examiner	Art Unit	
	Jacob Petranek	2183	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *18 July 2005*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *16-31* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *16-31* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *18 July 2005* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☒ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☒ All b)☐ Some * c)☐ None of:

1.☒ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date *9/27/2004*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 16-31 are pending.

2. The office acknowledges the following papers:

Claims, Oath, Drawings, Abstract, and specification filed on 7/18/2005.

Priority

3. The effective filing date for the subject matter defined in the pending claims in this application is 3/27/2003.

IDS

4. The information disclosure statement filed 9/27/2004 fails to comply with 37 CFR 1.98(a)(2), which requires a legible copy of each cited foreign patent document; each non-patent literature publication or that portion which caused it to be listed; and all other information or that portion which caused it to be listed. It has been placed in the application file, but the information referred to therein has not been considered.

5. The US Patent citation appears to have either cited the wrong inventor/applicant or wrong patent number. There isn't an inventor/applicant named Matu associated with the patent number given.

Drawings

6. The Examiner contends that the drawings submitted on 7/18/2005 are acceptable for examination proceedings.

Specification

7. The disclosure is objected to because of the following informalities:

8. The lengthy specification has not been checked to the extent necessary to determine the presence of all possible minor errors. The Applicant's cooperation is requested in correcting any errors of which the Applicant may become aware.

9. Appropriate correction is required.

Claim Objections

10. Claims 16, 18, 21-24, and 26-30 are objected to because of the following informalities:

11. Claim 16 recites "N+1 P-P (parallel program, hereinafter "P-P") module" at line 1 that should be changed to "[[N+1 P-P (parallel program, hereinafter "P-P")]] A N+1 parallel program (P-P) module" to establish the acronym "P-P".

12. Claim 16 recites "N+1 P-P branch programs module (N>=1)" at line 3 that should be changed to "N+1 P-P branch programs [[module (N>=1)]] module, where N is greater or equal to 1,".

13. Claim 16 recites "the P-P data" at line 6 that should be changed to "[[the]] a P-P data" for proper antecedent basis and to provide consistent claim language throughout the claims.

14. Claim 16 recites "the suspension status" at line 9 that should be changed to "[[the]] suspension status" for proper antecedent basis and to provide consistent claim language throughout the claims.

15. Claim 18 recites "the exit" at line 2 that should be changed to "[[the]] an exit" for proper antecedent basis and to provide consistent claim language throughout the claims.

16. Claim 21 recites "the call permission" at line 2 that should be changed to "[[the]] a call permission" for proper antecedent basis and to provide consistent claim language throughout the claims.

17. Claim 22 recites "the tokens … the transmissions … the synchronization" at lines 4-5 that should be changed to "[[the]] tokens … [[the]] transmissions … [[the]] synchronization" for proper antecedent basis and to provide consistent claim language throughout the claims.

18. Claim 23 recites "N+1 P-P branch programs" at line 1 that should be changed to "N+1 parallel program (P-P) branch programs" to establish the acronym "P-P".

19. Claim 23 recites "branch programs (N>=1)" at line 1 that should be changed to "branch [[programs (N>=1)]] programs, where N is greater or equal to 1,".

20. Claim 23 recites "the P-P" at line 4 that should be changed to "[[the]] a P-P" for proper antecedent basis and to provide consistent claim language throughout the claims.

21. Claim 23 recites "the ON/OFF switch" at line 4 that should be changed to "[[the]] an ON/OFF switch" for proper antecedent basis and to provide consistent claim language throughout the claims.

22. Claim 23 recites "the record" at line 7 that should be changed to "[[the]] record" for proper antecedent basis and to provide consistent claim language throughout the claims.

23. Claim 23 recites "the suspension status" at line 9 that should be changed to "[[the]] a suspension status" for proper antecedent basis and to provide consistent claim language throughout the claims.

24. Claim 23 recites "the reasons" at line 14 that should be changed to "[[the]] reasons" for proper antecedent basis and to provide consistent claim language throughout the claims.

25. Claim 23 recites "the exiting" at line 15 that should be changed to "[[the exiting]] an exit" for proper antecedent basis and to provide consistent claim language throughout the claims.

26. Claim 24 recites "the flag zone" at line 2 that should be changed to "[[the]] a flag zone" for proper antecedent basis and to provide consistent claim language throughout the claims.

27. Claim 26 recites "the flag" at line 4 that should be changed to "[[the]] a flag" for proper antecedent basis and to provide consistent claim language throughout the claims.

28. Claim 26 recites "the flag "consistency valid"" at line 12 that should be changed to "[[the]] a flag "consistency valid"" for proper antecedent basis and to provide consistent claim language throughout the claims.

29. Claim 26 recites "the consistency P-P branch program" at line 12 that should be changed to "[[the]] a consistency P-P branch program" for proper antecedent basis and to provide consistent claim language throughout the claims.

30. Claim 26 recites "the consistency operation" at line 13 that should be changed to "[[the]] a consistency operation" for proper antecedent basis and to provide consistent claim language throughout the claims.

31. Claim 27 recites "the "consistency valid" flag" at line 3 that should be changed to "[[the]] a "consistency valid" flag" for proper antecedent basis and to provide consistent claim language throughout the claims.

32. Claim 27 recites "the consistency operations" at lines 4-5 that should be changed to "[[the]] consistency operations" for proper antecedent basis and to provide consistent claim language throughout the claims.

33. Claim 27 recites "the N bits "had been read invalid" flag" at line 6 that should be changed to "[[the]] a N bits"had been read invalid" flag" for proper antecedent basis and to provide consistent claim language throughout the claims.

34. Claim 27 recites "the "had been invalid in consistency" flag" at line 17 that should be changed to "[[the]] a "had been invalid in consistency" flag" for proper antecedent basis and to provide consistent claim language throughout the claims.

35. Claim 27 recites "the returning port" at line 22 that should be changed to "[[the]] a returning port" for proper antecedent basis and to provide consistent claim language throughout the claims.

36. Claim 28 recites "the "read valid" flag" at line 4 that should be changed to "[[the]] a "read valid" flag" for proper antecedent basis and to provide consistent claim language throughout the claims.

37. Claim 28 recites "the N bits" at line 10 that should be changed to "[[the]] a N bits" for proper antecedent basis and to provide consistent claim language throughout the claims.

38. Claim 29 recites "P-P call instruction" at line 1 that should be changed to " parallel program (P-P) call instruction" to establish the acronym "P-P".

39. Claim 29 recites "the entry addresses" at line 1 that should be changed to "[[the]] entry addresses" for proper antecedent basis and to provide consistent claim language throughout the claims.

40. Claim 29 recites "the operations" at line 5 that should be changed to "[[the]] operations" for proper antecedent basis and to provide consistent claim language throughout the claims.

41. Claim 29 recites "the exiting" at line 7 that should be changed to "[[the]] exiting" for proper antecedent basis and to provide consistent claim language throughout the claims.

42. Claim 30 recites "the suspension status" at line 2 that should be changed to "[[the]] a suspension status" for proper antecedent basis and to provide consistent claim language throughout the claims.

Claim Rejections - 35 USC § 101

43. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

44. Claim 31 is rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter. Specifically, claim 31 includes no explicit recitation of any hardware component, nor does the claim include any component which must be interpreted solely as hardware. According to the published specification, paragraphs 9-10, the claimed components can be implemented in hardware, software, or a combination thereof. Consequently, when all of the components of claim 31 are software, applicant's claims are directed to software per se (for instance, instructions in a program), which does not fall into one of the four statutory categories of invention. Software is merely an abstract idea.

Claim Rejections - 35 USC § 112

45. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

46. Claims 16-28 and 31 are rejected under 35 U.S.C. §112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

Claim 16 recites "time division (without the need of time division operation under the structure of parallel computers)," and "sequence-net instructions (or subroutines)." Both of these limitations contain parenthesis with what is believed to be comments by the examiner. It's unclear if these comments are intended to further limit the claims or if they were intended as just commenting on the claims. For examination purposes, the limitations in the parenthesis will not be examined, and should either be deleted or amended into the claims to further limit the claims.

Claim 22 recites "P-P instructions (or subroutines)." Both citations of these limitations contain parenthesis with what is believed to be comments by the examiner. It's unclear if these comments are intended to further limit the claims or if they were intended as just commenting on the claims. For examination purposes, the limitations in the parenthesis will not be examined, and should either be deleted or amended into the claims to further limit the claims.

Claim 23 recites "if the check result is that all P-P branch programs are in suspended status …", "finding out the reasons …", and "processing the exiting …" It's unclear if the applicant intended that the finding and processing steps only occur when the check result shows that all P-P branch programs are suspended or if theses steps always occur regardless of the check result step. Figure 3a shows that these steps only occur when the check result shows that all P-P branch programs are suspended. For

examination purposes, proceeding with the finding and processing step only occur when all programs are in suspended status. An amendment that added the term "then" to the end of the "if the check result is that all P-P branch programs are in suspended status …" limitation and tabbed in the finding and processing limitations would clear up the confusion.

Claim 23 recites "if the check result is that more than one P-P branch program is in ready status …", "selecting … ready status", "selecting … parameters, and", and "running …" It's unclear if the applicant intended that the selecting and running steps only occur when the check result shows more than one program is ready or if these steps always occur regardless of the check result step. Figure 3a appears to support the former. It's also unclear if when only one program is in ready status, then the claims intended no action to occur as currently claimed. Figure 3a appears to support taking action when one or more programs is in ready status. For examination purposes, the check result determines if "one or more" programs is ready and when this occurs, the selecting and running steps occur. An amendment that added the term "then" to the end of the "if the check result is that more than one P-P branch program is in ready status …" limitation and tabbed in the selecting and running limitations, as well as changing "more than one" to "one or more" would clear up the confusion.

Claim 26 recites "if the flag is valid (indicating …)" and "the status bit … ready status." First, it's unclear if the limitations contained in parenthesis are supposed to be further limiting or are just general comments about the limitation. For examination purposes, the limitations in the parenthesis will not be examined, and should either be

deleted or amended into the claims to further limit the claims. Second, it's unclear if the applicant intended that the changing of the status bit occurs only when the flag is valid or if the status bit is always changed. For examination purposes, proceeding with the changing the status bit step only occurs when the flag is valid. An amendment that added the term "then" to the end of the "if the flag is valid" limitation and tabbed in changing the status bit limitation would clear up the confusion.

Claim 27 recites "if the consistency valid" flag", "setting the …", "proceeding the …", and "exiting the …" It's unclear if these three statements proceeding the flag checking step always occur or if they occur conditionally based on the status of the flag. Figure 3c appears to support the latter. For examination purposes, the three proceeding limitations occur conditionally based on the if statement. An amendment adding the term "then" and tabbing in the three proceeding limitations would clear up the confusion.

Claim 28 recites "if the "real valid" flag of the data is valid", "then reading the data ", and "returning the read data subroutine." It's unclear if the applicant intended that if the two steps proceeding the if statement always occur or if the two proceeding steps are conditional on the if statement. Figure 3d appears to support the latter. For examination purposes, the two proceeding steps after the if statement are dependent upon the data being valid. An amendment that added the term "then" to the end of the "if the "real valid" flag …" limitation and tabbed in the two proceeding limitations would clear up the confusion.

Claim 28 recites "if the "real valid" flag of the data is invalid", "in accordance …", "keeping the current …", and "the P-P branch program …" It's unclear if the applicant intended that if the three steps proceeding the if statement always occur or if the three proceeding steps are conditional on the if statement. Figure 3d appears to support the latter. For examination purposes, the three proceeding steps after the if statement are dependent upon the data being invalid. An amendment that added the term "then" to the end of the "if the "real valid" flag …" limitation and tabbed in the three proceeding limitations would clear up the confusion.

Claim 31 recites "P-P data sequence (in the N+1th branch program)." The limitation contains parenthesis with what is believed to be comments by the examiner. It's unclear if these comments are intended to further limit the claims or if they were intended as just commenting on the claims. For examination purposes, the limitations in the parenthesis will not be examined, and should either be deleted or amended into the claims to further limit the claims.

47. Claims 17-21, 24-25, are rejected due to their dependency.

Claim Rejections - 35 USC § 103

48. The following is a quotation of 35 U.S.C. §103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

49. Claims 23 and 28-30 are rejected under 35 U.S.C. §103(a) as being unpatentable over Nation et al. (U.S. 6,233,599).

50. As per claim 23:

Nation disclosed an operational method of N+1 P-P branch programs (N>=1) based on single machine environment, which comprises the following procedures:

initializing the N+1 P-P branch programs (Nation: Figure 1a elements 50 and 60, column 10 lines 16-22)(It's obvious to one of ordinary skill in the art that the threads are initialized when they are initially created.),

determining whether the P-P is terminated, and checking the ON/OFF switch flag of each P-P branch program (Nation: Figures 1a and 3d elements 50 and 86, column 9 lines 58-67)(It's obvious to one of ordinary skill in the art that when a thread is to be scheduled, that the valid indicators are to be checked to see if there are any valid threads that can be processed.), wherein:

> if a switch flag of all P-P branch programs is OFF, which indicates the P-P is terminated, the record for showing the termination of the P-P and the manner for connecting with external programs are processed (Nation: Figures 1a and 3d elements 50 and 86, column 9 lines 58-67)(It's obvious to one of ordinary skill in the art that when a thread is to be scheduled, that the valid indicators are to be checked to see if there are any valid threads that can be processed. If all threads are invalid, then all processes scheduled by the OS have been terminated and the processor is in a wait state until the OS schedules another process.),

once the switch flag of one P-P branch program is ON (Nation: Figures 1a and 3d elements 50 and 86, column 9 lines 58-67)(The valid bit indicates that the thread is active.), the suspension status for branch programs is checked (Nation: Figures 1a and 3a elements 50 and 86, column 9 lines 16-25 and column 11 lines 45-62)(When the controller attempts to schedule another thread, it must find an valid thread in a ready state.), wherein,

if the check result is that all P-P branch programs are in suspension status, which indicates that though the execution of some P-P branch programs is not finished, the P-P branch programs can not be executed (Nation: Figures 1a and 3a elements 50 and 86, column 9 lines 16-25 and column 11 lines 45-62)(It's obvious to one of ordinary skill in the art that if all active threads aren't ready to be scheduled, then no thread can be executed.),

finding out the reasons that the P-P branch programs can not be executed (Nation: Figure 3d element 89, column 9 lines 58-67)(The thread status registers detail the reason the thread was switched out.),

processing the exiting of the P-P branch programs based on the reasons (Nation: Figure 3d elements 86 and 89, column 9 lines 58-67)(It's obvious to one of ordinary skill in the art that an active thread can finish and become an invalid status. Thus, the reason for a thread switch would be the thread finishing execution.),

if the check result is that more than one P-P branch program is in ready status, enter the P-P branch programs queuing modules (Nation: Figures 1a and 3a elements 50 and 86, column 9 lines 16-25 and column 11 lines 45-62)(If there are multiple threads in ready status for execution, then it's obvious to one of ordinary skill in the art that the thread controller selects one based on priority.),

selecting a P-P branch program which is in ready status (Nation: Figure 1a element 50, column 11 lines 45-62)(The controller selects a valid thread that's ready for execution and loads all thread data needed into the processor.),

selecting a P-P branch program, loading its parameters (Nation: Figure 1a element 50, column 11 lines 45-62)(The controller selects a valid thread that's ready for execution and loads all thread data needed into the processor.), and

running the P-P branch program which is selected (Nation: Figure 1a element 50, column 11 lines 45-62)(The controller selects a valid thread that's ready for execution and loads all thread data needed into the processor.).

51. As per claim 28:

Nation disclosed the P-P operation method of Claim 23, wherein the P-P branch program comprises a read data subroutine, which comprises the following procedures:

checking the "read valid" flag of the data (Nation: Figure 1a element 16, column 6 lines 48-56)(It's obvious to one of ordinary skill in the art that a load instruction will check a cache line to see if it contains valid data.),

if the "read valid" flag of the data is valid,

then reading the data,

returning the read data subroutine (Nation: Figure 1a element 16, column 6 lines 48-56)(It's obvious to one of ordinary skill in the art that a load instruction will check a cache line to see if it contains valid data. If valid data exists, the data is returned to the processor's registers.),

if the "read valid" flag of the data is invalid (Nation: Figure 1a element 16, column 6 lines 48-56)(It's obvious to one of ordinary skill in the art that a load instruction will check a cache line to see if it contains valid data. If it's invalid, then the processor attempts to fetch from a lower-level cache.),

in accordance with the branch program of the read data subroutine, the N bits "had been read invalid" flags of are set as "had been read invalid" respectively (Nation: Figure 3d element 89, column 9 lines 58-67)(It's obvious to one of ordinary skill in the art that a thread switch can occur for a cache miss. Thus, a flag is written to the thread register indicating a cache miss caused the thread switch.),

keeping the current P-P branch program in suspension status, saving the current running situation of the P-P branch program so that the current P-P branch program may be continued from the suspension point (Nation: Figure 3d

element 89, column 9 lines 58-67 and column 11 lines 45-62)(It's obvious to one of ordinary skill in the art that a thread switch can occur for a cache miss. Thus, the thread's state is saved and a new threads state is loaded.), and

the P-P branch program exits and turns to branch suspension & exit module in order to re-select a new P-P branch program (Nation: Figure 1a element 50, column 11 lines 45-62)(It's obvious to one of ordinary skill in the art that a thread switch can occur for a cache miss. Thus, the thread controller changes to a different thread.).

52. As per claim 29:

Nation disclosed a P-P call-instruction including the entry addresses of N+1 P-P branch programs, which comprises:

a call-instruction, for calling a P-P entering & exit management module which is used to process and acquire the entry addresses of the parallel call of the branch programs (Nation: Figure 1a elements 62-66, column 11 lines 45-62)(Official notice is given that parent threads can create and destroy child threads in a program being executed. Thus, it's obvious to one of ordinary skill in the art that a parent thread can create and call a child thread to execute a subset of a process. The process involves a call to the start of execution in the child thread.),

a time-division-managing module for managing the operations of multiple branch programs, wherein the time-division-managing module selects and runs a first branch program from the multiple branch programs (Nation: Figures 1a and 3a elements 50 element 84, column 9 lines 16-25 and column 11 lines 45-62)(The thread controller is

the time-division managing module that manages the thread switching operation from one thread to another. The current thread executing is indicated by element 84 in figure 3a.), manages the exiting and returning of the first branch program (Nation: Figure 1a element 50, column 11 lines 59-62)(The thread controller manages exiting from a current thread and starting execution in another thread.), and selects and runs a second branch program from the multiple branch programs (Nation: Figure 1a element 50, column 11 lines 59-62)(The thread controller selects the next thread to execute.); the time-division-managing module returns to the P-P entering & exit management module until all the multiple branch programs is run (Nation: Figure 1a element 50, column 11 lines 45-62)(It's obvious to one of ordinary skill in the art that the processor continues execution until all threads have completed execution.).

53. As per claim 30:

Nation disclosed the time-division-managing module of Claim 29, wherein the time-division managing module (Nation: Figures 3a and 3d elements 80 and 85) manages the "suspension status" (Nation: Figure 3d element 87, column 9 lines 58-67)(The not-ready flag indicates a suspension status.), "ready status" (Nation: Figure 3d element 87, column 9 lines 58-67)(The ready flag indicates a ready status.), and "running status" (Nation: Figure 3a element 84, column 9 lines 16-25)(The active thread ID field indicates the current running thread.) of the multiple branch programs in response to return information from the N+I P-P branch programs in suspension status (Nation: Figures 3a and 3d elements 80 and 85, column 9 lines 6-14 and lines 51-58)(

The thread status registers are updated during program execution to indicate a current status of all of the threads.).

54. Claim 31 is rejected under 35 U.S.C. §103(a) as being unpatentable over Nikhil et al. (U.S. 5,499,349).

55. As per claim 31:

Nation disclosed a combination structure of P-P call-instructions and call-permission-instructions, which comprises:

a precondition for executing the P-P call-instructions is that the call-permission-instructions are executed first (Nikhil: Figure 8 elements 60-64, column 7 lines 52-65 and column 8 lines 52-61)(The fork instruction reads upon the call-permission instruction that calls to a new thread to execute. It's obvious to one of ordinary skill in the art that further fork or start instructions can occur in elements 62 or 64 that create a third thread or start a new second thread. These instructions read upon the other call-instruction that is executed after the first fork.),

the call-instructions and the call-permission-instructions are located in different branch programs respectively (Nikhil: Figure 8 elements 60-64, column 7 lines 52-65 and column 8 lines 52-61)(The fork instruction reads upon the call-permission instruction that calls to a new thread to execute. It's obvious to one of ordinary skill in the art that further fork or start instructions can occur in elements 62 or 64 that create a third thread or start a new second thread. These instructions read upon the other call-

instruction that is executed after the first fork. The instructions are contained in different threads.), and

the call-permission-instructions and the P-P data sequence are sequenced simultaneously (Nikhil: Figure 8 element 60, column 7 lines 52-65)(The fork instruction is sequenced simultaneously with data associated with the instruction.).

56. Claims 16-22 and 24-25 are rejected under 35 U.S.C. §103(a) as being unpatentable over Nation et al. (U.S. 6,233,599), in view of Nikhil et al. (U.S. 5,499,349).

57. As per claim 16:

Nation disclosed a N+1 P-P (parallel program, hereinafter "P-P") module based on a single machine environment, which comprises:

N+1 P-P branch programs module (N>=1), which is run by operating the N+I P-P branch programs which have an object code independent structures, in the way of time division (Nation: Figure 1a elements 50 and 62-66, column 11 lines 45-62)(The multiple threads of execution read upon the branch programs and are executed by way of time division by switching threads.), for making a transmission and consistency of the P-P data under the supports of three classes of sequence-net instructions for reading P-P data, writing P-P data (Nation: Figure 1a element 30, column 6 lines 25-35)(Load and store instructions read and write data in the threads. They support transmission of data and consistency among programs by reading and writing data back to main memory.); and

a managing program module (Nation: Figures 3a and 3d elements 80 and 85), for supporting the suspension status (Nation: Figure 3d element 87, column 9 lines 58-67)(The not-ready flag indicates a suspension status.), ready status (Nation: Figure 3d element 87, column 9 lines 58-67)(The ready flag indicates a ready status.), and running status (Nation: Figure 3a element 84, column 9 lines 16-25)(The active thread ID field indicates the current running thread.) of the P-P branch programs in response to information from the P-P branch programs (Nation: Figures 3a and 3d elements 80 and 85, column 9 lines 6-14 and lines 51-58)(The thread status registers are updated during program execution to indicate a current status of all of the threads.).

Nation failed to teach a third class of sequence-net instructions for making P-P data consistency in said P-P branch programs.

However, Nikhil disclosed a third class of sequence-net instructions for making P-P data consistency in said P-P branch programs (Nikhil: Figure 9 element 76, column 8 lines 8-31)(The join instruction allows for synchronization between the two threads by having the first thread wait until the second executes the join instruction. The synchronization allows for consistent results by ensuring that program execution doesn't continue before it's supposed to.).

Nation disclosed the process of switching between threads when a current thread is stalled, but lacks the information about how threads are created in the first place. One of ordinary skill in the art would have been motivated by this lack of detail to find the Nikhil reference that details the use of fork and join instructions to create and destroy threads. Thus, it would have been obvious to one of ordinary skill in the art to

implement the fork and join instructions into the processor of Nation to detail how threads are created and destroyed.

58. As per claim 17:

Nation and Nikhil disclosed the N+1 P-P module of claim 16, further comprising a suspension-processing module, for processing the suspension status, ready status and running status of the current P-P branch programs (Nation: Figures 3a and 3d elements 80 and 85, column 9 lines 6-14 and lines 51-58)(It's obvious to one of ordinary skill in the art that logic exists for changing the thread flags when needed.).

59. As per claim 18:

Nation and Nikhil disclosed the N+1 P-P module of claim 16, further comprising a P-P entering & exit management module, for initializing the P-P for each application and processing the exit of the P-P in response to information from a time-division-managing module (Nation: Figure 1a element 50, column 11 lines 45-62)(The thread controller processes thread switching to cause threads to enter and exit from a currently executing status.).

60. As per claim 19:

Nation and Nikhil disclosed the N+ 1 P-P module of claim 16, wherein the N+1th P-P branch program of P- P module executes a P-P data sequence, which is represented by a data consistency operation (Nikhil: Figure 9 element 76, column 8 lines 8-31)(The join instruction allows for synchronization between the two threads by having the first thread wait until the second executes the join instruction. The synchronization allows for consistent results by ensuring that program execution doesn't

continue before it's supposed to. The instructions executed prior to the join instruction consist of a sequence.).

61. As per claim 20:

Nation and Nikhil disclosed the N+1 P-P module of claim 19, wherein a token consistency operation of N+1th branch program corresponds to the P-P data, which is made consistency one by one directly by the control of the N+1th program (Nikhil: Figure 9 element 76, column 8 lines 8-31)(The join instruction allows for synchronization between the two threads by having the first thread wait until the second executes the join instruction, which allows for consistent results by ensuring correct execution. The join instruction in the parent thread reads upon the token consistency operation.), wherein if the P-P data has been written by other P-P branch program, then the consistency of the P-P data is executed (Nikhil: Figure 9 element 76, column 8 lines 8-31)(If the child thread executes the join instruction first, then when the parent thread executes the join instruction program execution can continue.), otherwise the N+1th branch program is in suspension status (Nikhil: Figure 9 element 76, column 8 lines 8-31)(If the parent thread executes the join instruction first, then when the parent thread waits until the join instruction is executed in the child.), after the execution of the P-P data's consistency, the P-P data read by other P- P branch program is valid (Nikhil: Figure 9 element 76, column 8 lines 8-31)(Once both threads execute the join, data can be read by the remaining thread to continue execution.).

62. As per claim 21:

Nation and Nikhil disclosed the N+1 P-P module of claim 20, wherein, when P-Ps are embed and child P-Ps are called, the call-permission of the child P-P and the P-P data sequence are sequenced in the N+1th branch program (Nikhil: Figure 8 element 60, column 7 lines 52-65)(A parent thread can call a child thread via a fork instruction. A join instruction can be used by the parent thread to finish the child thread.).

63. As per claim 22:

Nation and Nikhil disclosed the N+1 P-P module of claim 20, wherein, the N+1 programs include three kinds of P-P instructions for reading the data, writing the data (Nation: Figure 1a element 30, column 6 lines 25-35)(Load and store instructions read and write data in the threads. They support transmission of data and consistency among programs by reading and writing data back to main memory.), and making the data in consistency respectively, and the P-P instructions have the capability for setting, detecting, and processing the tokens, and also have the capability for supporting the transmissions of P-P data and the synchronization of P-P branch programs (Nikhil: Figure 9 element 76, column 8 lines 8-31)(The join instruction reads upon the final kind of P-P instruction. It allows for synchronization between the two threads by having the first thread wait until the second executes the join instruction. The synchronization allows for consistent results by ensuring that program execution doesn't continue before it's supposed to.).

64. As per claim 24:

Nation disclosed the N+1 P-P operation method of Claim 23.

Nation failed to teach the initializing procedure comprises the loading of parameters of P-P branch programs and the resetting of the flag zone the P-P.

However, Nikhil disclosed the initializing procedure comprises the loading of parameters of P-P branch programs and the resetting of the flag zone the P-P (Nikhil: Figure 8 element 60, column 7 lines 50-65)(Nation: Figure 1a element 60)(The fork instruction allows for the creation of a new thread off of a parent thread. It's obvious to one of ordinary skill in the art that the new thread will have a thread context loaded into element 60 of Nation upon the creation and will result in setting the appropriate flags.).

Nation disclosed the process of switching between threads when a current thread is stalled, but lacks the information about how threads are created in the first place. One of ordinary skill in the art would have been motivated by this lack of detail to find the Nikhil reference that details the use of fork and join instructions to create and destroy threads. Thus, it would have been obvious to one of ordinary skill in the art to implement the fork and join instructions into the processor of Nation to detail how threads are created and destroyed.

65. As per claim 25:

Nation and Nikhil disclosed the N+1 P-P operation method of Claim 24, wherein, the loading of the parameters of the P-P branch program comprises:

loading the flag ON for the N+1 branch programs of the P-Ps (Nation: Figure 3d element 86)(Nikhil: Figure 8 element 60, column 7 lines 50-65)(Creation of a new thread results in setting the valid indicator in the individual thread status register.), setting an entry address for each P-P branch programs, setting a data initial values for respective

registers (Nikhil: Figure 8 element 60, column 7 lines 50-65)(It's obvious to one of ordinary skill in the art that the creation of the new thread also creates a PC for the thread to begin program execution at in the PC register.), and resetting a P-P flag zone and a data flag of the P-P branch program (Nation: Figure 3d elements 86-88)(It's obvious to one of ordinary skill in the art that the valid, ready, and priority fields are set for a new thread upon creation.).

66. Claims 26-27 are rejected under 35 U.S.C. §103(a) as being unpatentable over Nation et al. (U.S. 6,233,599), in view of Meyer et al. (U.S. 7,640,315).

67. As per claim 26:

Nation and Meyer disclosed the N+1 P-P operation method of Claim 23, wherein the P-P branch program comprises a subroutine for writing data, which comprises the following procedures:

performing the writing operation for a P-P data (Meyer: Figure 1b, column 1 lines 49-58)(The test and set operation attempts to write data.),

checking the flag "had been invalid for consistency" of the P-P data (Meyer: Figure 1b, column 1 lines 49-58)(The lock variable state is tested.),

if the flag is valid,

the status bit of N+1th P-P branch program of the ready/suspension flag of P-P is changed from the suspension status to the ready status (Meyer: Figure 1b, column 1 lines 59-66)(Nation: Figure 1a element 50, column 11 lines 45-62)(Official notice is given that thread switches can occur based on a timeout value. Thus, the thread can

become suspended after a number of attempts to lock the variable. Later, the thread will successfully lock the variable and will be changed from a suspended state to a ready state.),

if the flag is invalid, no specific operation is executed (Meyer: Figure 1b, column 1 lines 49-58)(The test and set instruction isn't executed if the variable is locked.),

establishing the flag "consistency valid" and allowing the consistency P-P branch program to perform the consistency operation to the data (Nation: Figures 1a and 3d elements 50 and 89, column 9 lines 58-67 and column 11 lines 45-62)(Official notice is given that thread switches can occur based on a timeout value. Thus, the thread can become suspended after a number of attempts to lock the variable. Therefore, the thread switch identifier flag can indicate a timeout caused the thread switch. The thread switch allows for other threads to finish execution that causes the lock variable to be unlocked.), and

exiting write data subroutine (Meyer: Figure 1b, column 1 lines 49-58)(The test and set instruction isn't executed if the variable is locked and the routine is exited.).

The advantage of using semaphores is that they can be used to allow code to execute atomically without other processor's reading and writing to a section of data (Meyer: Column 1 lines 42-48). One of ordinary skill in the art would have been motivated by this advantage to implement locking variables in the processor of Nation. Thus, it would have been obvious to one of ordinary skill in the art at the time of the invention to implement the process of locking variables in the processor of Nation for the advantage of correctly synchronizing programs.

68. As per claim 27:

Nation and Meyer disclosed the P-P operation method of Claim 23, wherein the P-P branch program comprises a consistency data subroutine, which comprises the following procedures:

checking the "consistency valid" flag of this data (Meyer: Figure 1b, column 1 lines 49-58)(The lock variable state is tested.), wherein,

if the "consistency valid" flag of this data is valid, the consistency operations of data and token are executed (Meyer: Figure 1b, column 1 lines 59-66)(If the variable is successfully locked, then the critical section can be entered and executed.),

checking whether the N bits "had been read invalid" flag of this data is valid (Nation: Figure 1a element 16, column 6 lines 48-56)(It's obvious to one of ordinary skill in the art that a load instruction will check a cache line to see if it contains valid data. It's obvious to one of ordinary skill in the art that the valid bit of a cache line can be a plurality of bits.),

if the "had been read invalid" flag is valid, then the relevant P-P branch programs are changed to the ready status from the suspension status based on the content of the flag "had been read invalid" (Nation: Figures 1 and 3d elements 16 and 89, column 9 lines 58-67)(It's obvious to one of ordinary skill in the art that a thread switch can occur for a cache miss. Thus, a flag is written to the thread register indicating a cache miss caused the thread switch. Upon return, the thread's status is changed to ready and the cache data has since been loaded into the cache and is available.),

if the "had been read invalid" flag of this data is invalid, no specific operation is proceeded, the data consistency subroutine is terminated and returned (Nation: Figure 3d element 89, column 9 lines 58-67)(It's obvious to one of ordinary skill in the art that a thread switch can occur for a cache miss. Thus, a flag is written to the thread register indicating a cache miss caused the thread switch.),

if the "consistency valid" flag of this data is invalid, the P-P branch program is suspended (Nation: Figures 1a and 3d elements 50 and 89, column 11 lines 45-62)(Official notice is given that thread switches can occur based on a timeout value. Thus, the thread can become suspended after a number of attempts to lock the variable. This causes the thread to become suspended/inactive.),

setting the "had been invalid in consistency" flag (Nation: Figures 1a and 3d elements 50 and 89, column 9 lines 58-67 and column 11 lines 45-62)(Official notice is given that thread switches can occur based on a timeout value. Thus, the thread can become suspended after a number of attempts to lock the variable. Therefore, the thread switch identifier flag can indicate a timeout caused the thread switch.),

proceeding the suspension process of the current N+1th branch program, saving the current running situation of the N+1th P-P branch program to be used when returning the P-P branch program (Nation: Figure 1a element 50, column 11 lines 45-62)(The thread controller saves the current thread's state prior to switching to another thread.), and

exiting the P-P branch program, and keeping the P-P branch program in suspension status and quitting from the returning port, in order to re-select a new P-P branch program (Nation: Figure 1a element 50, column 11 lines 45-62)(The thread controller selects a new thread upon a thread switch.).

The advantage of using semaphores is that they can be used to allow code to execute atomically without other processor's reading and writing to a section of data (Meyer: Column 1 lines 42-48). One of ordinary skill in the art would have been motivated by this advantage to implement locking variables in the processor of Nation. Thus, it would have been obvious to one of ordinary skill in the art at the time of the invention to implement the process of locking variables in the processor of Nation for the advantage of correctly synchronizing programs.

Conclusion

The following is text cited from 37 CFR 1.111(c): In amending in reply to a rejection of claims in an application or patent under reexamination, the applicant or patent owner must clearly point out the patentable novelty which he or she thinks the claims present in view of the state of the art disclosed by the references cited or the objections made. The applicant or patent owner must also show how the amendments avoid such references or objections.

The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

Dubey et al. (U.S. 5,812,811), taught instruction threads with forking and inter-thread communication.

Safranek et al. (U.S. 6,493,809), taught maintaining order of write operations for memory consistency.

McKenney (U.S. 2002/0194436), taught software implementation of memory barriers.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Jacob Petranek whose telephone number is 571-272-5988. The examiner can normally be reached on M-F 8:00-4:30.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on (571) 272-4162. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

/Jacob Petranek/
Examiner, Art Unit 2183